eunix: whoami

Eric Bailey

November 29, 2017 [1]

[1] Last updated November 29, 2017

A reimplementation of `whoami` for my own edification.

Contents

The `main` Function

1b ⟨*Define the* `main` *function.* 1b⟩≡

```
int main(int argc, char *argv[])
{
    ⟨Process given options. 3c⟩

    ⟨Print the user name associated with the current effective user ID. 3d⟩

    return 0;
}
```

This code is used in chunk 1a.
Defines:
`argc`, used in chunk 3c.
`argv`, used in chunk 3c.
`main`, never used.

1a ⟨* 1a⟩≡

```
⟨Include headers. 2a⟩

⟨Forward declarations. 3a⟩

⟨Define the main function. 1b⟩

⟨Define the usage function. 3b⟩
```

Root chunk (not used in this document).

Include Headers

Include the core input and output functions from the C standard library.

2a ⟨*Include headers.* 2a⟩≡
```
#include <stdio.h>
```
This definition is continued in chunk 2.
This code is used in chunk 1a.
Defines:
`EOF`, used in chunk 3c.
`printf`, used in chunks 3b and 4c.

Describe sys/types.h

2b ⟨*Include headers.* 2a⟩+≡
```
#include <sys/types.h>
```
This code is used in chunk 1a.
Defines:
`uid_t`, used in chunk 3d.

Describe pwd.h

2c ⟨*Include headers.* 2a⟩+≡
```
#include <pwd.h>
```
This code is used in chunk 1a.
Defines:
`getpwuid`, used in chunk 4b.
`passwd`, used in chunk 3d.

Describe unistd.h

2d ⟨*Include headers.* 2a⟩+≡
```
#include <unistd.h>
```
This code is used in chunk 1a.
Defines:
`geteuid`, used in chunk 3e.

Include the GNU `getopt` function from the GNU C Library.

"The `getopt` function gets the next option argument from the argument list specified by the `argv` and `argc` arguments. Normally these values come directly from the arguments received by `main`." – GNU, 2017

2e ⟨*Include headers.* 2a⟩+≡
```
#include <getopt.h>
```
This code is used in chunk 1a.
Defines:
`getopt`, used in chunk 3c.

The usage Function

Define the usage function, which displays information about how to use whoami.

3a ⟨*Forward declarations.* 3a⟩≡
```
void usage();
```
This code is used in chunk 1a.
Uses usage 3b.

3b ⟨*Define the* usage *function.* 3b⟩≡
```
void usage()
{
    printf("Usage: whoami\n");
}
```
This code is used in chunk 1a.
Defines:
usage, used in chunk 3.
Uses printf 2a.

Processing Options

If any optiosn are given, complain about the first one (via getopt), print the usage information, and return a nonzero status code.

3c ⟨*Process given options.* 3c⟩≡
```
if (getopt(argc, argv, "") != EOF) {
    usage();
    return 1;
}
```
This code is used in chunk 1b.
Uses argc 1b, argv 1b, EOF 2a, getopt 2e, and usage 3b.

Printing the Current User's Name

Describe the variables

3d ⟨*Print the user name associated with the current effective user ID.* 3d⟩≡
```
struct passwd *pw;
uid_t uid;
uid_t NO_UID = -1;
```

This definition is continued in chunks 3 and 4.
This code is used in chunk 1b.
Defines:
NO_UID, used in chunk 4a.
pw, used in chunk 4.
uid, used in chunks 3 and 4.
Uses passwd 2c and uid_t 2b.

Get the effective user ID.

3e ⟨*Print the user name associated with the current effective user ID.* 3d⟩+≡
```
uid = geteuid();
```

This code is used in chunk 1b.
Uses geteuid 2d and uid 3d.

Describe this check

4a ⟨*the user ID is* NO_UID 4a⟩≡
```
uid == NO_UID
```
This code is used in chunk 4c.
Uses NO_UID 3d and uid 3d.

Search the user database for an entry with a matching uid.

Describe getpwuid and what it means to fail

4b ⟨*find a user with a matching* uid 4b⟩≡
```
pw = getpwuid(uid)
```
This code is used in chunk 4c.
Uses getpwuid 2c, pw 3d, and uid 3d.

If ⟨*the user ID is* NO_UID 4a⟩ or we're unable to [[⟨*find a user with a matching* uid 4b⟩, print a descriptive error message and return a nonzero status code.

4c ⟨*Print the user name associated with the current effective user ID.* 3d⟩+≡
```
if (⟨the user ID is NO_UID 4a⟩ || !(⟨find a user with a matching uid 4b⟩)) {
    printf("Cannot find name for user ID %lu\n",
           (unsigned long int) uid);
    return 1;
}
```
This code is used in chunk 1b.
Uses printf 2a and uid 3d.

4d ⟨*Print the user name associated with the current effective user ID.* 3d⟩+≡
```
puts(pw->pw_name);
```
This code is used in chunk 1b.
Uses pw 3d.

Full Listing

```c
#include <stdio.h>
#include <sys/types.h>
#include <pwd.h>
#include <unistd.h>
#include <getopt.h>


void usage();


int main(int argc, char *argv[])
{
    if (getopt(argc, argv, "") != EOF) {
        usage();
        return 1;
    }

    struct passwd *pw;
    uid_t uid;
    uid_t NO_UID = -1;

    uid = geteuid();

    if (uid == NO_UID || !(pw = getpwuid(uid))) {
        printf("Cannot find name for user ID %lu\n",
               (unsigned long int) uid);
        return 1;
    }
    puts(pw->pw_name);

    return 0;
}


void usage()
{
    printf("Usage: whoami\n");
}
```

Chunks

⟨* 1a⟩ 1a
⟨*Define the* `main` *function.* 1b⟩ 1a, 1b
⟨*Define the* `usage` *function.* 3b⟩ 1a, 3b
⟨*find a user with a matching* `uid` 4b⟩ 4b, 4c
⟨*Forward declarations.* 3a⟩ 1a, 3a
⟨*Include headers.* 2a⟩ 1a, 2a, 2b, 2c, 2d, 2e
⟨*Print the user name associated with the current effective user ID.* 3d⟩ 1b, 3d, 3e, 4c, 4d
⟨*Process given options.* 3c⟩ 1b, 3c
⟨*the user ID is* `NO_UID` 4a⟩ 4a, 4c

Index

References

GNU. The GNU C Library: Using the getopt function. https://www.gnu.org/software/libc/manual/html_node/Using-Getopt.html, 2017. Accessed: 2017-11-05.

To-Do